Newcomb

A Solar System Ephemeris Program

Marc A. Murison
murison@riemann.usno.navy.mil

James L. Hilton
hil@ham.usno.navy.mil

Astronomical Applications Department
U. S. Naval Observatory
Washington, D.C.

April 23, 1997

Table of Contents

Chapter 1: Newcomb Project Outline and Top Level Program Structure

1.1. Introduction.

This chapter describes the top-level structure of the planned Naval Observatory Solar System Ephemeris program, called Newcomb. Newcomb is intended to be the successor of, and derives its inheritance more or less from, PEP, the Planetary Ephemeris Program at the Smithsonian Astrophysical Observatory.[1] Computer program design and language capabilities have advanced far beyond the anticipations of more than three decades ago when PEP was written. That, combined with the practical inability to add further significant capabilities or modifications to PEP, has been deemed sufficient cause for development of a new ephemeris program. Additional motivations are that it is to the USNO's great advantage to have a complete capability in-house, and that Newcomb will provide a check against the JPL DE program (as well as against PEP).

Chief among the advantages of writing a new program is the opportunity to make use of object-oriented design (OOD) and object-oriented programming (OOP). Newcomb will be written in C++ and, for the graphical user interface, in java. We will take full advantage of standard OOP/OOD concepts and techniques, including data encapsulation, template classes, polymorphism, and multiple inheritance. The benefits of a completely object-oriented approach are many, including faster prototyping and development, fewer and more easily locatable coding errors, vastly simpler and more intuitive design, more sophisticated functionality, easily extensible architecture, and (most importantly) drastically reduced maintenance costs. Another major benefit is that the program can be up and running with minimal functionality, allowing further capability to be easily and painlessly incorporated as need arises.

Ease of extensibility is largely a result of object-oriented design, but it is also directly related to how good that design is. Hence, considerable effort is going into the design of Newcomb. Experience in the software industry abundantly shows that the payoff later on in terms of maintenance and extensibility is far out of proportion to the effort expended early on — in the design stages — of the program life cycle.

This chapter discusses some major design issues and, more importantly, presents an initial framework for further development.

1.2. Project Outline.

In this beginning part of the Newcomb project, tasks naturally fall into three main categories: program design, documentation, and science applications. A rough outline of the most obvious subjects that must be addressed early on is:

- I. Design Issues
  - A. numerical integration scheme
    - 1. object-oriented design
    - 2. Integrable objects have knowledge of dynamical environment as well as the ability to dynamically evolve in that environment.
  - B. exception handling
    - 1. all exceptions fully recoverable
    - 2. procedure stack traceback
  - C. robust parameter estimation
    - 1. Singular Value Decomposition (SVD)
    - 2. swipe a package from elsewhere
  - D. graphical user interface

[1]Information on PEP may be found at http://cfa-www.harvard.edu/~reasen/ssd.html

1. use GUI application frameworks package (such as ZAF from Zinc) to ensure platform portability
E. reduction of observations
1.
2.
F. individual class design and testing

II. Science Issues and Projects to Consider
A. asteroids
1. masses from orbital interactions
2. provide ephemerides (services to the community)
3. cumulative effects on planetary motions
i. Asteroids are the largest source of "noise" in the orbits of Mars, Earth-Moon system.
B. lunar motion
1. chaotic dynamics
i. predictions from numerical models
ii. comparisons with LLR data
2. radiation pressure **[ref]**
3. resonant interaction between tidal and GR terms
4. lunar librations
C. Nordtvedt η parameter (anomalous gravitational field energy effects — i.e., a difference between gravitational and inertial mass proportional to the gravitational binding energy of a body)
D. GR precession
1. lunar orbit
2. Earth's spin
E. bounds on time variation of the gravitational constant
F. millisecond pulsars
1. derive Earth orbit
G. bounds on dark matter in the solar system?
H. planetary satellites?
1. centroiding vs. satellite-derived center of mass
I. other science?

III. Documentation
A. code
1. source documentation model (see TM96-01)
2. interface ("user's manual")
B. algorithms
C. physics
1. GR and partial derivatives
D. parameter estimation and error and correlation analysis
E. numerical integration design
F. reduction of observations

1.3. Top Level Structure.

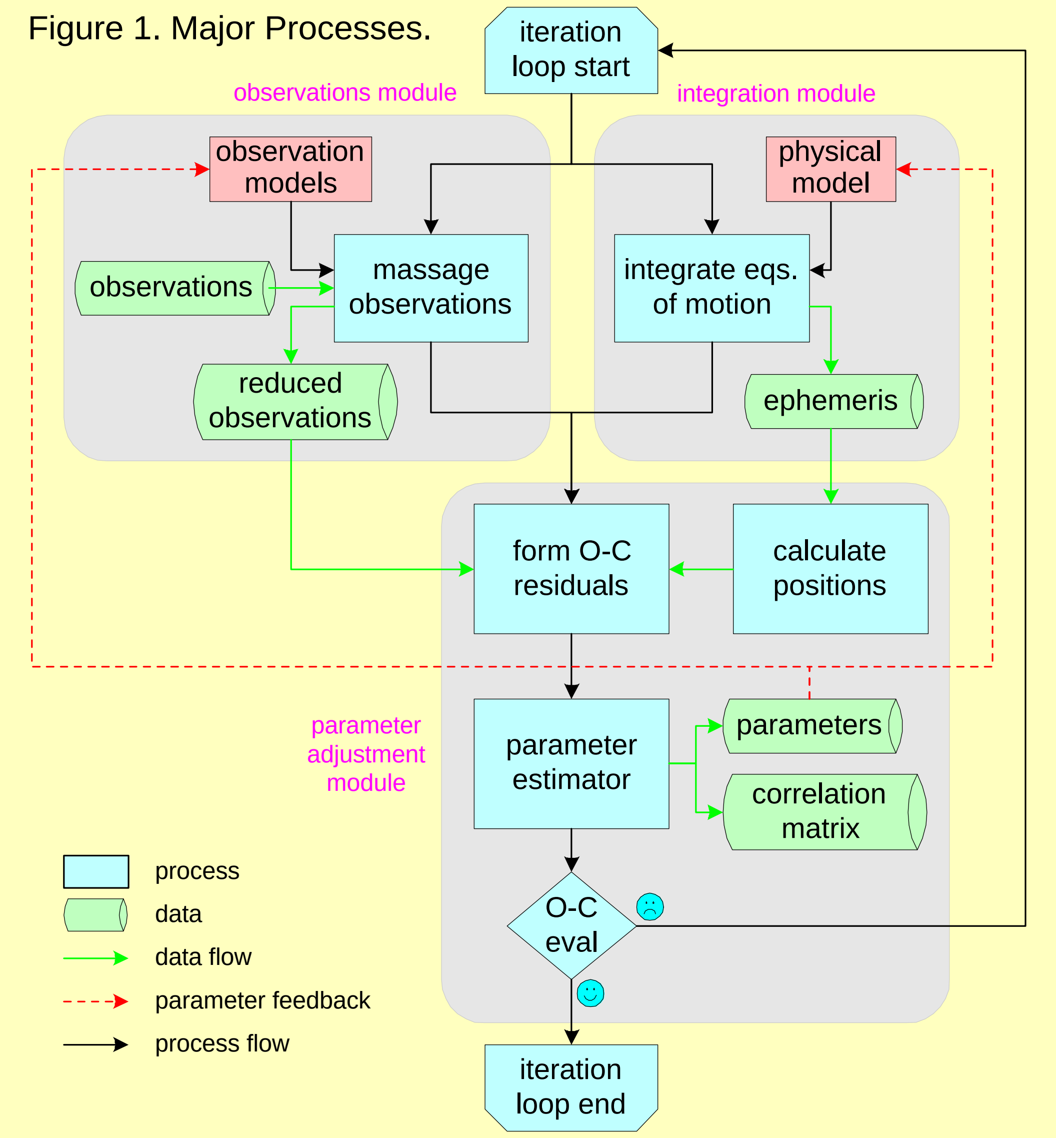
The top level process structure of Newcomb is shown in Figure 1. Basic operation is as follows. The observations module is responsible for reading input astrometric observations and "massaging" them as necessary. Massaging operations are listed in Chapter 2. The observations will be of various types (cf. Figure 30), taken at various observing locations (cf. Figure 2), including spacecraft.

The integration module is responsible for numerically integrating a sophisticated dynamical model of the solar system — including general relativistic terms, a detailed Earth-Moon system, planetary spin vectors including precession and nutation, and an unlimited number of asteroids — to produce an ephemeris.

The model ephemeris is then compared with the observations in the O-C section of the parameter adjustment module to produce a set of residuals. The parameter estimator uses the partial derivatives of the model equations with respect to the model parameters (including initial conditions) to solve in a least squares sense for the most probable set of model parameter values that minimizes the O-C residuals. The adjusted model parameters are then fed back into both the ephemeris generator and the observation transformation methods. The data are rereduced as necessary, and a new ephemeris is generated. These are again combined to produce a new set of residuals. This process is iterated until the residuals satisfy predetermined success criteria.

At the end of the iterative process, we will have produced an ephemeris that best fits the observations, given the model used, as well as the best-fit model parameters, formal error estimates of those parameters, and the parameter cross-correlations. The parameter error estimates and parameter correlations are derived from the partial derivatives and the correlation matrix from the least squares analysis. Experience with PEP has shown that, normally, at most only a few iterations are needed.

Figure 1. Major Processes.



Chapter 2: The Observations Module

2.1. Introduction.

Perhaps the most difficult section of the program will be the module that processes input observations and reduces them to a form suitable for passage to the O-C section of the parameter adjustment module (see Figure 1). Essentially, the observations will be sent to the O-C section in the form of apparent positions, corrected for various biases, including (but not limited to):

- catalog corrections
- delay/doppler bias corrections
- coordinate frame fiducialization
- aberration corrections
- nutation and precession

Integral to this section are the specific types of observational datasets and the specific types of observational platforms. The data and platform types vary widely.

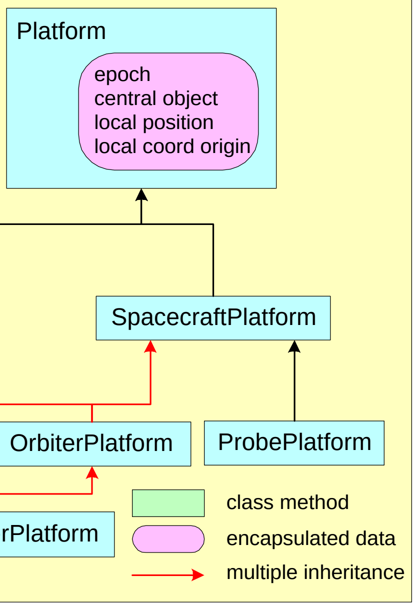
2.2. Observing Platforms.

One must consider the various observing platforms presently available in the solar system. They are

I. Planet
    A. Earth
        1. Earth-based observatories
        2. Earth orbiters
    B. Planetary landers
    C. Planetary orbiters
II. Deep space probes (i.e., gravitationally unbound from all planets and satellites)

Figure 2 shows the object hierarchy of observing platforms.[2] The C++ code classes will reflect this hierarchy. Each input datastream will contain relevant observing platform information. An appropriate observing platform object will encapsulate this information. Each type of platform object also encapsulates the necessary functionality (referred to as *method*) to provide information needed to manipulate or transform data of the corresponding type (see Figure 3 on page 8). For example, planetary observing platform objects know how to precess and nutate coordinates to a specified epoch. Each base class contains parameters and functionality common to all subclasses derived from it. The derived classes contain only the additional or specialized parameters and functionality required to handle platforms of a specific kind. For example, since all planetary platforms have a basic precession and nutation capability, these methods reside in the base class PlanetPlatform. An EarthPlatform object automatically inherits all the functionality and data of PlanetPlatform. The EarthPlatform object therefore contains only additional abilities, data, or refinements, for example precession parameters specific to the Earth. Proper use of inheritance

Figure 2. Platform Class Hierarchy.



[2]Arrows in Figures 2 and 3 point *from* derived classes *to* parent (also called *base*) classes. This is the standard notation.

eliminates code duplication for common tasks in a natural and intuitive way. The inheritance mechanism is built into the C++ language and therefore requires no enforcement by or special discipline from the programmer.

Figure 2 intentionally shows only the major class types, in accord with the introductory nature appropriate to this Chapter. It is a simple matter to derive further specialized classes from the base classes shown. For example, one would derive a VikingOrbiter from OrbiterPlatform.

2.3. Observation Types.

The various observation data types fall naturally into the two broad categories, timing (in a sense, the radial coordinate from the observer) and positions (on the sky, i.e. transverse to the radial direction). The complete breakdown is as follows:

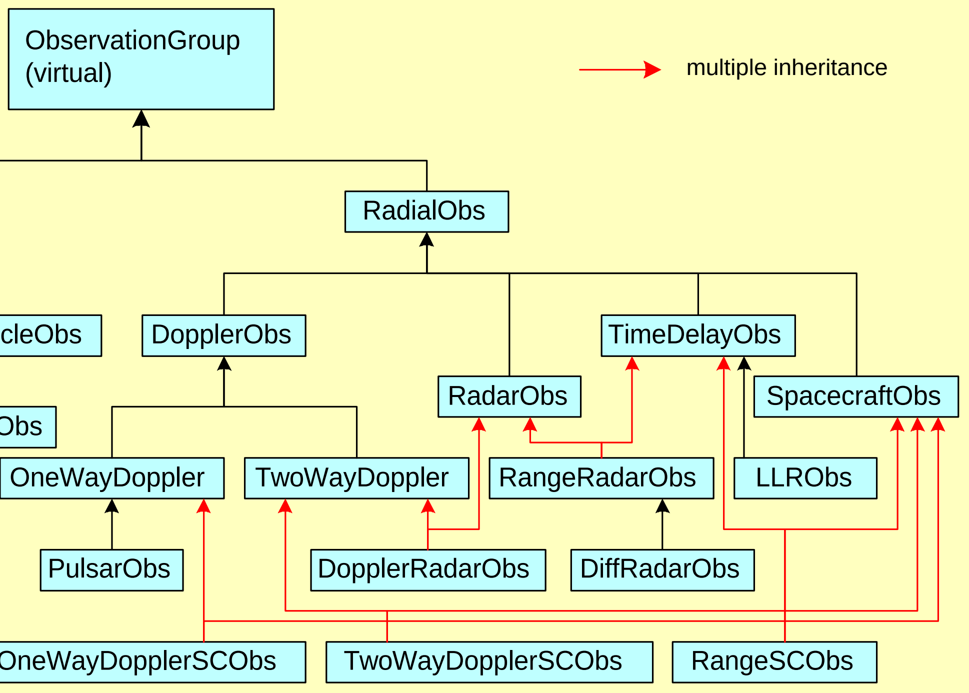
I. Transverse (position)
  A. Optical observations
    1. Global positions
      i. Transit circle
    2. Differential positions
    3. Occultations
      i. Satellite-planet
      ii. Star-planet
      iii. Spacecraft-planet
    4. Transits
      i. Solar
      ii. Planetary
  B. VLBI
II. Radial (timing)
  A. Doppler observations
    1. One-way
      i. Pulsars
      ii. Spacecraft
    2. Two-way
      i. Radar
      ii. Spacecraft
  B. Time delay observations
    1. LLR
    2. Radar
      i. Differential radar
      ii. Radar closure
    3. Spacecraft
      i. Single
      ii. Multi

For reasons having mainly to do with datasets that are currently insufficiently large or insufficiently accurate to have a substantial effect on ephemeris accuracy, early versions of Newcomb will not include the observation types shown in light red. Because extensibility is built into the design of Newcomb, adding further capabilities as they become necessary will involve minimal effort — there is no need, from a maintenance standpoint, to include capabilities that are anticipated to go unused for a long time. That is, with a good object oriented design we do not have to worry so much about "making room" for anticipated future capabilities. Figure 3 shows the proposed class hierarchy.

Each type of input data stream will contain embedded type information, and instantiations of the appropriate data objects will handle the data. The specific objects shown in Figure 3 encapsulate not only the corresponding observational data but also the functionality required to reduce that data type. For example, notice that all datatype objects have, via inheritance from the base class Observation, platform information and the ability to handle (say) aberration.

As with Figure 2, Figure 3 is intentionally not complete, especially regarding encapsulated data and method details. However, all the important base classes, and their inheritance dependencies, are shown.

Figure 3.
Observation Data
Class Hierarchy.

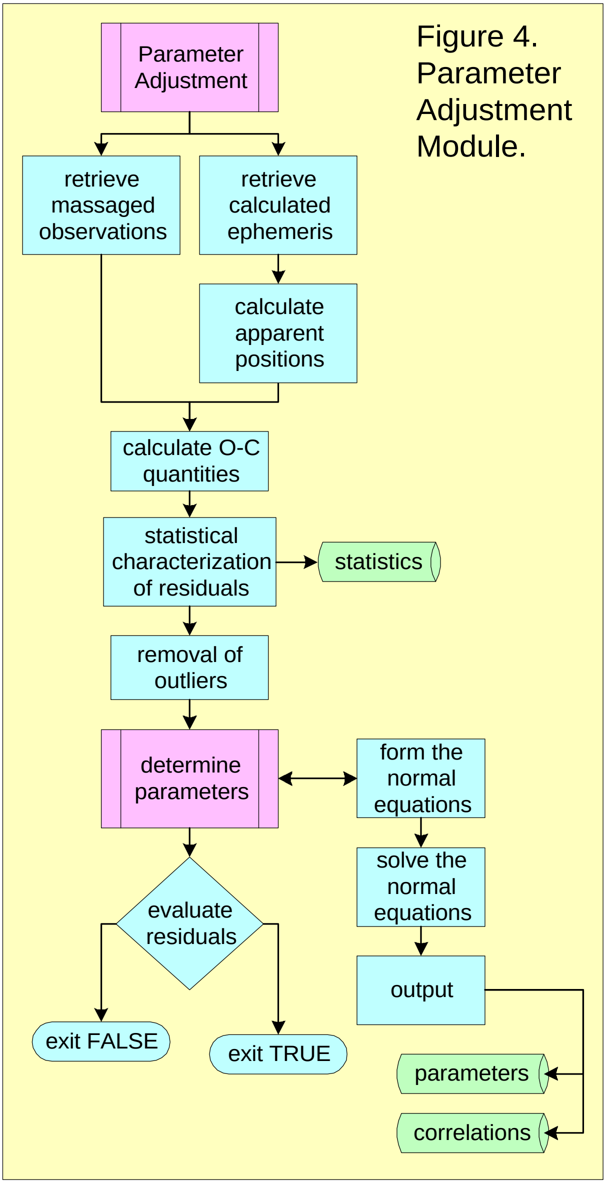


Chapter 3: The Parameter Adjustment Module

3.1. Overview.

The parameter adjustment module is relatively straightforward. The processed observations and the calculated ephemeris data are requested and then compared, forming the O-C residuals. First, coordinate frame compatibility between the observations and the synthetic ephemeris is reconciled. The calculated ephemeris must be transformed to apparent positions in order to match the observations. The residuals are characterized, with statistical and descriptive output going to disk as well as to an output window on-screen. At this point, outlying data points can be automatically detected and removed. The core of the module follows with the determination of parameters via a maximum likelihood estimator. The normal equations are formed and solved, and the parameters and associated formal error estimates are saved. Finally, the residuals are evaluated, and the module exits with a solution "acceptability" code. Figure 4 illustrates the process.

Matrix inversion is accomplished via singular value decomposition (SVD), which is very robust and offers useful diagnostics for ill-conditioned matrices. Singularities are automatically detected and corrected, and the problem parameters are identified. In essence, if the algorithm encounters an ill-conditioned matrix, it safely steps around the problem point(s) and proceeds in such a way as to mine the matrix for the maximum amount of information. When a singularity (rare in practice) or degenerate column (not rare!) is encountered, the combination of parameters that led to the fault is easily extracted. Thus, not only are singularities safely handled, but — more importantly — parameter combinations to which the data are insensitive are automatically identified. It is unusual to encounter a computational method that is this reliable and blowup-proof. I have already developed and tested matrix inversion using SVD and incorporated it into the Matrix utility class (Chapter ??). With regard to Newcomb, SVD is a "plug'n'play" capability.


Parameter Adjustment
retrieve massaged observations
retrieve calculated ephemeris
calculate apparent positions
calculate O-C quantities
statistical characterization of residuals
statistics
removal of outliers
determine parameters
form the normal equations
solve the normal equations
output
evaluate residuals
exit FALSE
exit TRUE
parameters
correlations
Figure 4. Parameter Adjustment Module.

3.2. Linear Parameter Estimation.

If the observational errors are uncorrelated, then maximum likelihood estimation becomes a simple least squares estimation. *We will assume that the observational errors are uncorrelated and normally distributed.* More precisely, the data errors are assumed to have two components — systematic errors and random errors. We assume the systematic components are modeled with bias parameters, which will be estimated. It is the random components which we assume to be uncorrelated and normally distributed. Hence, the probability of a specific data-set occurring, given a model with *n* physical and bias parameters $\boldsymbol{a}$ (a vector of length *n*), is the product of the probabilities of the *N* individual data points:

$$P \sim \prod_{i=1}^{N} \exp\left[-\tfrac{1}{2}\left(\frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i}\right)^2\right] \tag{1}$$

where y_i are the data and $y(t_i,\boldsymbol{a})$ is the model function. Maximizing the probability means minimizing the negative of the logarithm of this expression, which within a factor of two becomes

$$\chi^2 \equiv \sum_{i=1}^{N} \frac{[y_i - y(t_i,\boldsymbol{a})]^2}{\sigma_i^2} \tag{2}$$

We recognize minimization of (2) as being equivalent to minimizing chi-squared, the usual least squares method. For notational convenience, introduce the summation operator

$$\{\cdot\}_i \equiv \sum_{i=1}^{N}(\cdot) \tag{3}$$

Thus, (2) is

$$\chi^2 = \left\{\frac{[y_i - y(t_i,\boldsymbol{a})]^2}{\sigma_i^2}\right\}_i \tag{4}$$

When this operator is used, summation is always over the *N* observational data points.

Minimization of (4) requires the set of *n* equations

$$-\frac{1}{2}\frac{\partial\chi^2}{\partial\boldsymbol{a}} = \left\{\frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i}\frac{\partial y(t_i,\boldsymbol{a})}{\sigma_i\,\partial\boldsymbol{a}}\right\}_i = 0 \tag{5}$$

to be satisfied simultaneously. Let the model function have the generalized representation

$$y(t,\boldsymbol{a}) = \sum_{k=1}^{n} a_k Y_k(t) \equiv \boldsymbol{a}\cdot\boldsymbol{Y}(t) \tag{6}$$

where the $Y_k(t)$ are functions of time and the vector $\boldsymbol{Y}(t) \equiv [Y_1(t), Y_2(t), \ldots, Y_n(t)]$. The basis functions $Y_k(t)$ are not restricted to linearity and may have any form (polynomials, trig functions, etc.). The linearity that is important is in the dependence of the model function on the parameters $\boldsymbol{a}$. Then (5) becomes

$$\left\{\left[\frac{y_i}{\sigma_i} - \sum_{k=1}^{n} a_k \frac{Y_k(t_i)}{\sigma_i}\right]\frac{\boldsymbol{Y}(t_i)}{\sigma_i}\right\}_i = 0 \tag{7}$$

Define the vector

$$\boldsymbol{S} \equiv \left\{\frac{y_i}{\sigma_i}\boldsymbol{Z}(t_i)\right\}_i \tag{8}$$

and the symmetric matrix

$$A_{jk} \equiv \{Z_j(t_i)Z_k(t_i)\}_i \tag{9}$$

where, for convenience, we have set

$$\boldsymbol{Z}(t_i) \equiv \frac{\boldsymbol{Y}(t_i)}{\sigma_i} \tag{10}$$

Then (7) becomes

$$\sum_{k=1}^{n} A_{jk}\,a_k = S_j \tag{11}$$

or

$$\boldsymbol{A}\cdot\boldsymbol{a} = \boldsymbol{S} \tag{12}$$

Hence, the parameters are determined from the solution vector

$$\boldsymbol{a} = \boldsymbol{A}^{-1}\cdot\boldsymbol{S} \tag{13}$$

Equations (12) are the *normal equations*. The matrix $\boldsymbol{C} \equiv \boldsymbol{A}^{-1}$ is the *covariance matrix*. The covariance matrix is the key to *formal* knowledge of the errors in the parameter estimates, as well as the correlations between the various parameters. Indeed, as we shall see in the next section, the parameter errors are the diagonal elements of $\boldsymbol{C}$,

$$\sigma_k \equiv \sqrt{C_{kk}} \tag{14}$$

The off-diagonal elements are the cross correlations,

$$\sigma_{jk} \equiv \sqrt{C_{jk}} \tag{15}$$

3.3. Formal Parameter Errors.

This section contains a derivation of (14). We begin with a statement of propagation of errors, which we then use to define the formal errors of the parameters. Using the definition (9), we then arrive at (14).

Suppose we have a scalar function of M parameters, $f(p_1, \ldots, p_M)$. Then the variation of f is

$$\Delta f = \sum_{k=1}^{M} \frac{\partial f}{\partial p_k} \Delta p_k \equiv \frac{\partial f}{\partial \boldsymbol{p}} \cdot \Delta \boldsymbol{p} \tag{16}$$

Now suppose N measurements of the parameters $\boldsymbol{p}$ are taken and then the function f computed from these observationally determined parameters. The variance of f is then

$$\sigma_f^2 = \frac{1}{N} \{(f_i - \langle f \rangle)^2\}_i \tag{17}$$

Approximate $f_i - \langle f \rangle$ by Δf from (16). Then

$$\begin{aligned}
\sigma_f^2 &= \frac{1}{N}\{(\Delta f)^2\}_i = \frac{1}{N}\left\{\left(\frac{\partial f}{\partial \boldsymbol{p}} \cdot \Delta \boldsymbol{p}\right)^2\right\}_i \\
&= \frac{1}{N}\left\{\sum_{k=1}^{M}\left(\frac{\partial f}{\partial p_k}\right)^2 \Delta p_k^2 \right. \\
&\qquad \left. + \sum_{\substack{j,k \\ j \neq k}} \frac{\partial f}{\partial p_j}\frac{\partial f}{\partial p_k} \Delta p_j \Delta p_k \right\}_i \\
&= \sum_k \left(\frac{\partial f}{\partial p_k}\right)^2 \frac{1}{N}\{(\Delta p_k)^2\}_i \\
&\qquad + \sum_{\substack{j,k \\ j \neq k}} \frac{\partial f}{\partial p_j}\frac{\partial f}{\partial p_k}\frac{1}{N}\{\Delta p_j \Delta p_k\}_i \\
&\equiv \sum_k \left(\frac{\partial f}{\partial p_k}\right)^2 \sigma_k^2 + \sum_{\substack{j,k \\ j \neq k}} \frac{\partial f}{\partial p_j}\frac{\partial f}{\partial p_k}\sigma_{jk}^2
\end{aligned} \tag{18}$$

If the observations are uncorrelated, the cross terms (double sum) tend to cancel. Equation (18) describes the propagation of errors.

Now consider our least squares parameter estimation, eq. (13). The parameters $\boldsymbol{a}$ are quantities determined from N measurements, $[y_i]$, with corresponding measurement errors σ_i. In light of (18), and with a slight abuse of vector notation, we may write the parameter variance as

$$\sigma_{\boldsymbol{a}}^2 = \left\{\left(\frac{\partial \boldsymbol{a}}{\partial y_i}\right)^2 \sigma_i^2\right\}_i + \left\{\left\{\frac{\partial \boldsymbol{a}}{\partial y_i}\frac{\partial \boldsymbol{a}}{\partial y_j}\sigma_{ij}^2\right\}_j\right\}_i \tag{19}$$

We will assume that the observational errors are individually uncorrelated, $\sigma_{ij} = 0 \ \forall \ i \neq j$. Then we are left with

$$\sigma_{\boldsymbol{a}}^2 = \left\{\left(\frac{\partial \boldsymbol{a}}{\partial y_i}\right)^2 \sigma_i^2\right\}_i \tag{20}$$

Now, from (13) and (8) we have

$$\boldsymbol{a} = \boldsymbol{C} \cdot \boldsymbol{S} = \boldsymbol{C} \cdot \left\{\frac{y_i}{\sigma_i}\boldsymbol{Z}(t_i)\right\}_i \tag{21}$$

Thus,

$$\frac{\partial \boldsymbol{a}}{\partial y_i} = \frac{1}{\sigma_i}\boldsymbol{C} \cdot \boldsymbol{Z}(t_i) \tag{22}$$

Hence, (20) becomes

$$\sigma_{\boldsymbol{a}}^2 = \{(\boldsymbol{C} \cdot \boldsymbol{Z}(t_i))^2\}_i \tag{23}$$

The kth component of (23) may be written

$$\sigma_{a_k}^2 = \{[\boldsymbol{C} \cdot \boldsymbol{Z}(t_i)]_k [\boldsymbol{C} \cdot \boldsymbol{Z}(t_i)]_k\}_i \tag{24}$$

Since $\boldsymbol{C}$ is symmetric,

$$\sigma_{a_k}^2 = \{[\boldsymbol{C} \cdot \boldsymbol{Z}(t_i)]_k [\boldsymbol{Z}(t_i) \cdot \boldsymbol{C}]_k\}_i \tag{25}$$

Interchange the order of summations:

$$\sigma_{a_k}^2 = [\boldsymbol{C} \cdot \{\boldsymbol{Z}(t_i)\boldsymbol{Z}(t_i)\}_i \cdot \boldsymbol{C}]_{kk} \tag{26}$$

Using (9), we have

$$\begin{aligned}
\sigma_{a_k}^2 &= [\boldsymbol{C} \cdot \boldsymbol{A} \cdot \boldsymbol{C}]_{kk} \\
&= [\boldsymbol{C} \cdot \boldsymbol{C}^{-1} \cdot \boldsymbol{C}]_{kk} \\
&= [\boldsymbol{C}]_{kk}
\end{aligned} \tag{27}$$

which completes the derivation of (14).

3.4. Nonlinear Parameter Estimation.

Consider the condition equations (5). If the model function $y(t, \boldsymbol{a})$ is nonlinear in the parameters $\boldsymbol{a}$, then the simple separation, eq. (6), which led to the easily-solved normal equations, (12), is no longer possible. Our goal is to minimize (4) with respect to variation of the model parameters, despite the nonlinear dependence of the model function on the parameters. To do this, we will adopt an iterative approach. Let $\boldsymbol{a} \equiv \tilde{\boldsymbol{a}} + \Delta \boldsymbol{a}$, where $\tilde{\boldsymbol{a}}$ are the true (or, more accurately, best) values of the parameters. Assume we start with parameter values that are close to the best values. We can then approximate (4) with a truncated Taylor series:

$$\chi^2(\boldsymbol{a}) \approx \chi^2(\tilde{\boldsymbol{a}}) - \Delta \boldsymbol{a} \cdot \boldsymbol{B} + \tfrac{1}{2}\Delta \boldsymbol{a} \cdot \boldsymbol{M} \cdot \Delta \boldsymbol{a} \tag{28}$$

where

$$\boldsymbol{B} \equiv -\left.\frac{\partial\chi^2}{\partial \boldsymbol{a}}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \text{ and } [M]_{ij} \equiv \left.\frac{\partial\chi^2}{\partial a_i \partial a_j}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \qquad (29)$$

Within the paraboloidal approximation (28), the correction vector $\Delta\boldsymbol{a}$ which minimizes $\chi^2(\boldsymbol{a})$ is given by the value for which the parameter correction gradient vanishes:

$$\frac{\partial\chi^2(\boldsymbol{a})}{\partial\Delta\boldsymbol{a}} \approx -\boldsymbol{B} + \Delta\boldsymbol{a}\cdot\boldsymbol{M} \qquad (30)$$

Hence,

$$\Delta\boldsymbol{a} = \boldsymbol{M}^{-1}\cdot\boldsymbol{B} \qquad (31)$$

where we have used the fact that $\boldsymbol{M}$ is symmetric. For $\boldsymbol{B}$ and $\boldsymbol{M}$ we calculate

$$\begin{aligned} \boldsymbol{B} &= \left\{ \frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i^2} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial \boldsymbol{a}}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right\}_i \\ [M]_{jk} &= \left\{ \frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i^2} \left.\frac{\partial^2 y(t_i,\boldsymbol{a})}{\partial a_j \partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right. \\ &\quad \left. - \frac{1}{\sigma_i^2} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial a_j}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right\}_i \end{aligned} \qquad (32)$$

Unfortunately, although the merit function is unitless, the elements of eqs. (32) are not (in general). This can lead too easily to an ill-conditioned matrix $\boldsymbol{M}$ when the parameters exhibit numerically widely disparate units. Following Hessler *et al.*[3], let us redefine $\boldsymbol{B}$ and $\boldsymbol{M}$ in a unitless fashion:

$$\begin{aligned} B_k &\equiv -\tilde{a}_k \left.\frac{\partial\chi^2}{\partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \\ &= \tilde{a}_k \left\{ \frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i^2} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right\}_i \end{aligned} \qquad (33)$$

and

$$\begin{aligned} [M]_{jk} &\equiv \tilde{a}_j \tilde{a}_k \left.\frac{\partial\chi^2}{\partial a_j \partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \\ &= \tilde{a}_j \tilde{a}_k \left\{ \frac{y_i - y(t_i,\boldsymbol{a})}{\sigma_i^2} \left.\frac{\partial^2 y(t_i,\boldsymbol{a})}{\partial a_j \partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right. \\ &\quad \left. - \frac{1}{\sigma_i^2} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial a_j}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \left.\frac{\partial y(t_i,\boldsymbol{a})}{\partial a_k}\right|_{\boldsymbol{a}=\tilde{\boldsymbol{a}}} \right\}_i \end{aligned} \qquad (34)$$

To keep a unitless form for the merit function, define

$$\delta a_k \equiv \frac{a_k - \tilde{a}_k}{\tilde{a}_k} \qquad (35)$$

Using (33) and (34), we can rewrite (28) as

$$\chi^2(\boldsymbol{a}) \approx \chi^2(\tilde{\boldsymbol{a}}) - \delta\boldsymbol{a}\cdot\boldsymbol{B} + \tfrac{1}{2}\delta\boldsymbol{a}\cdot\boldsymbol{M}\cdot\delta\boldsymbol{a} \qquad (36)$$

Requiring

$$\frac{\partial\chi^2(\boldsymbol{a})}{\partial\,\delta\boldsymbol{a}} \approx -\boldsymbol{B} + \delta\boldsymbol{a}\cdot\boldsymbol{M} = 0 \qquad (37)$$

we have, finally,

$$\delta\boldsymbol{a} = \boldsymbol{M}^{-1}\cdot\boldsymbol{B} \qquad (38)$$

where $\boldsymbol{B}$ and $\boldsymbol{M}$ are given by (33) and (34). Notice that the form of eqs. (38) is identical to that of eqs. (13). The procedure, then, is to start with an initial set of values for the parameters, calculate the correction via eqs. (38), correct the parameter values, and repeat until the values stop changing significantly.

[3]J.P. Hessler, D.H. Current, and P.J. Ogren (1996), "A new scheme for calculating weights and describing correlations in nonlinear least-squares fits", *Computers in Physics* **10**, 186.

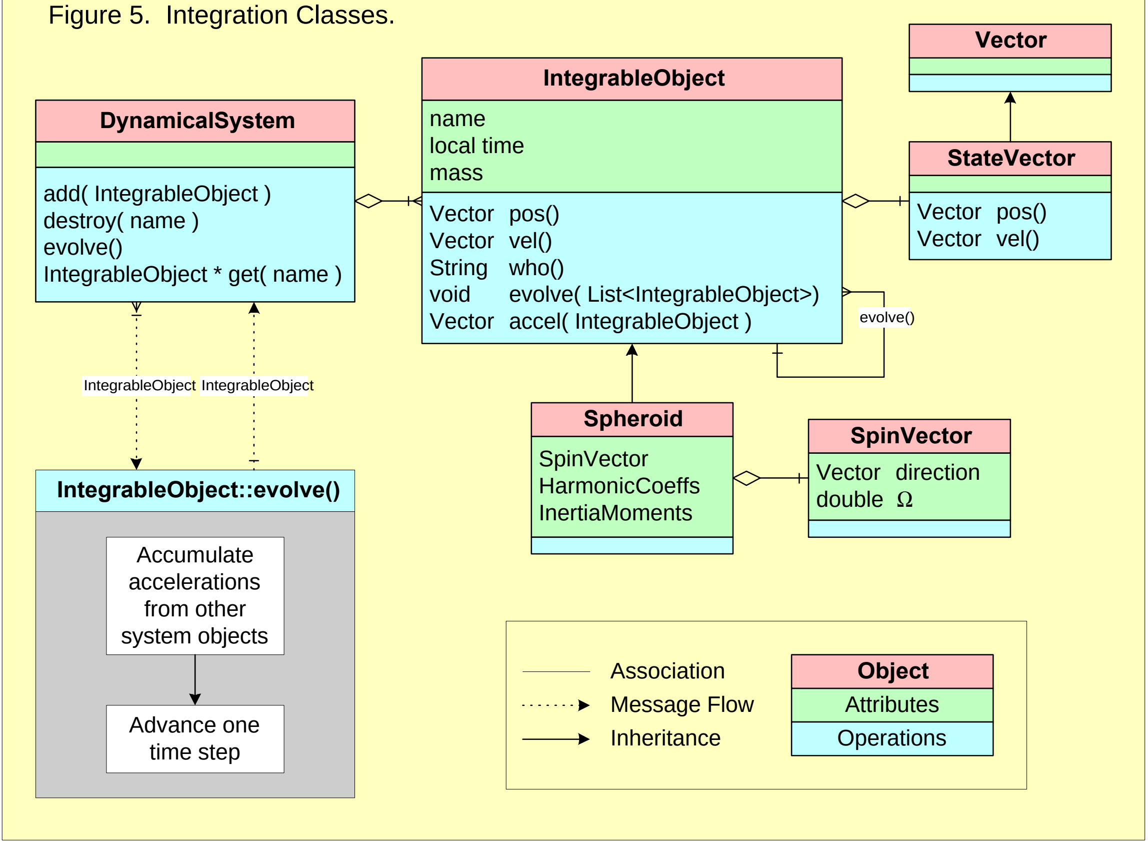
Chapter 4: The Integration Module

4.1. Physical Model.

4.2. Numerical Integration Algorithms.

4.3. An Object Oriented Approach.

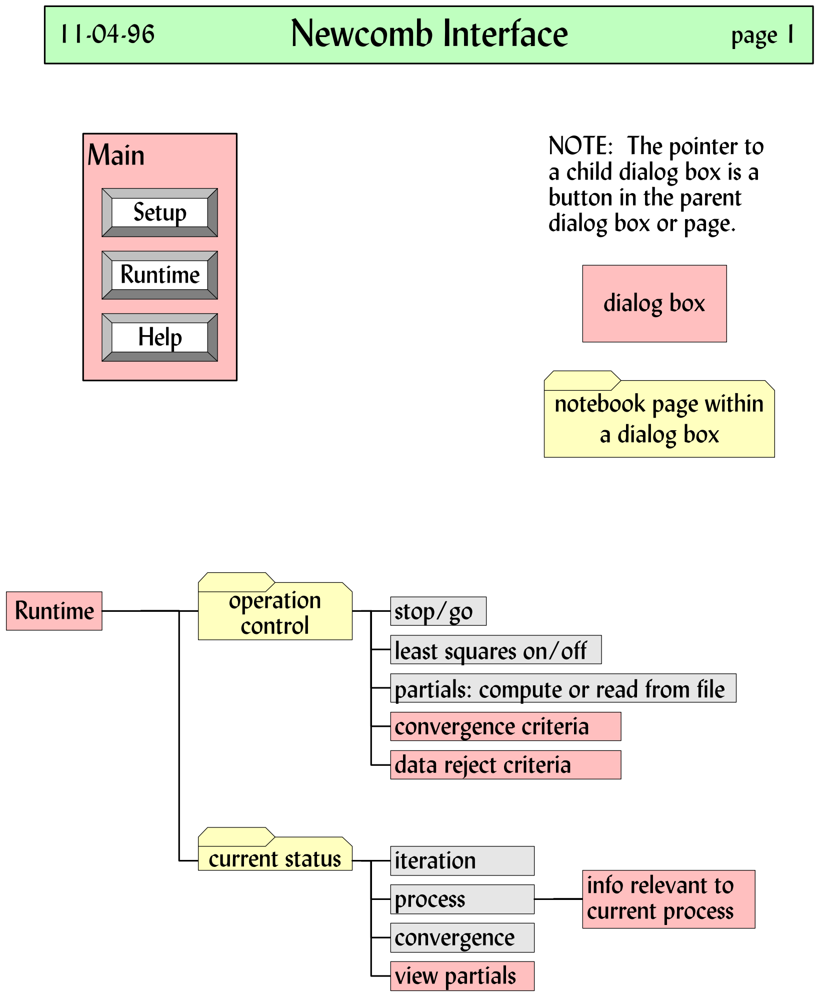
Figure 5. Integration Classes.

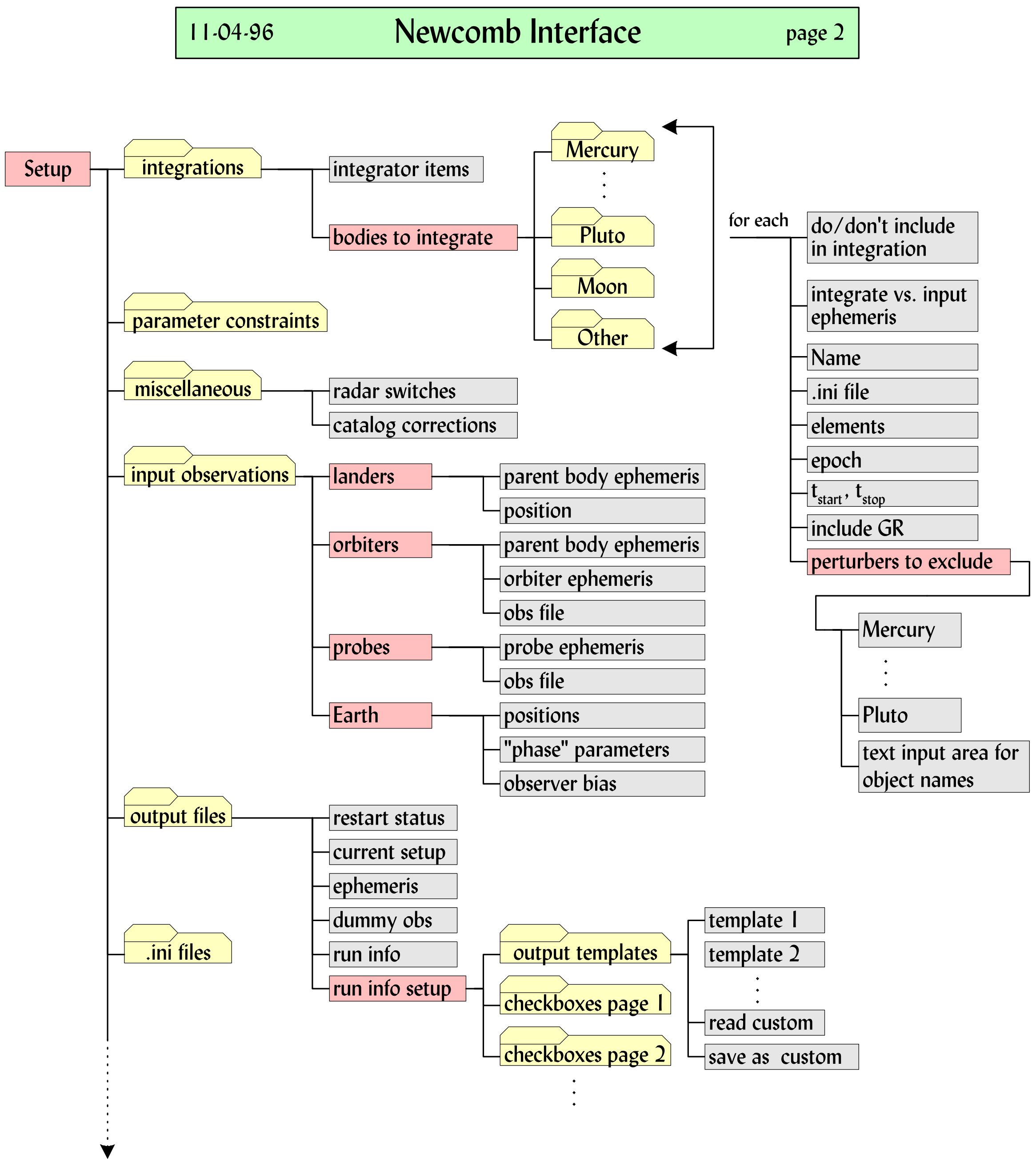


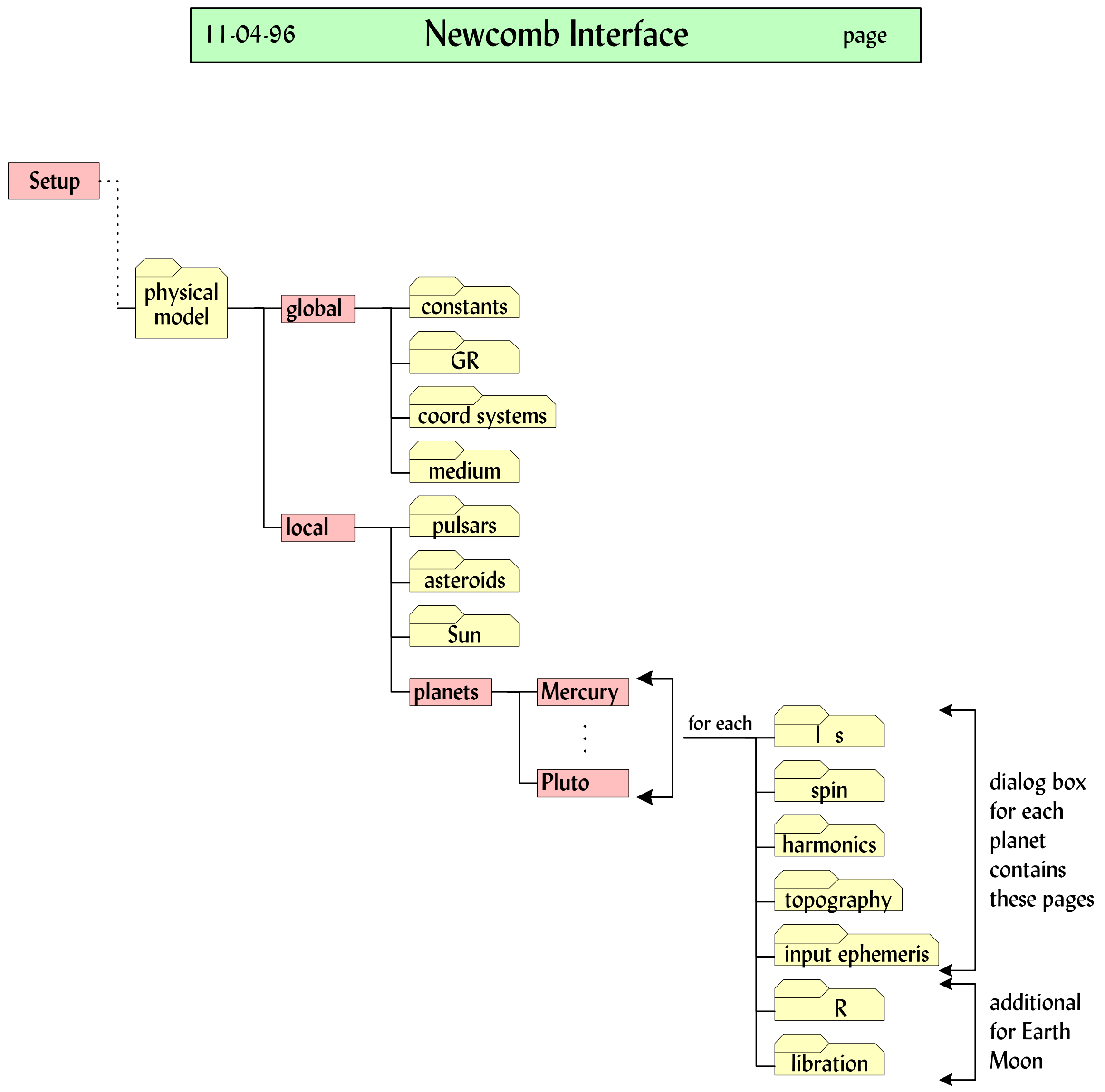
4.4. Ephemeris Generation.

Chapter 5: The User Interface

5.1. Introduction.




11-04-96
Newcomb Interface
page 2
Setup
integrations
integrator items
bodies to integrate
Mercury
Pluto
Moon
Other
for each
do/don't include in integration
integrate vs. input ephemeris
Name
.ini file
elements
epoch
t_{start}, t_{stop}
include GR
perturbers to exclude
Mercury
Pluto
text input area for object names
parameter constraints
miscellaneous
radar switches
catalog corrections
input observations
landers
parent body ephemeris
position
orbiters
parent body ephemeris
orbiter ephemeris
obs file
probes
probe ephemeris
obs file
Earth
positions
"phase" parameters
observer bias
output files
restart status
current setup
ephemeris
dummy obs
run info
run info setup
output templates
template 1
template 2
read custom
save as custom
checkboxes page 1
checkboxes page 2
.ini files


11-04-96
Newcomb Interface
page
Setup
physical model
global
constants
GR
coord systems
medium
local
pulsars
asteroids
Sun
planets
Mercury
Pluto
for each
I s
spin
harmonics
topography
input ephemeris
R
libration
dialog box for each planet contains these pages
additional for Earth Moon

Index